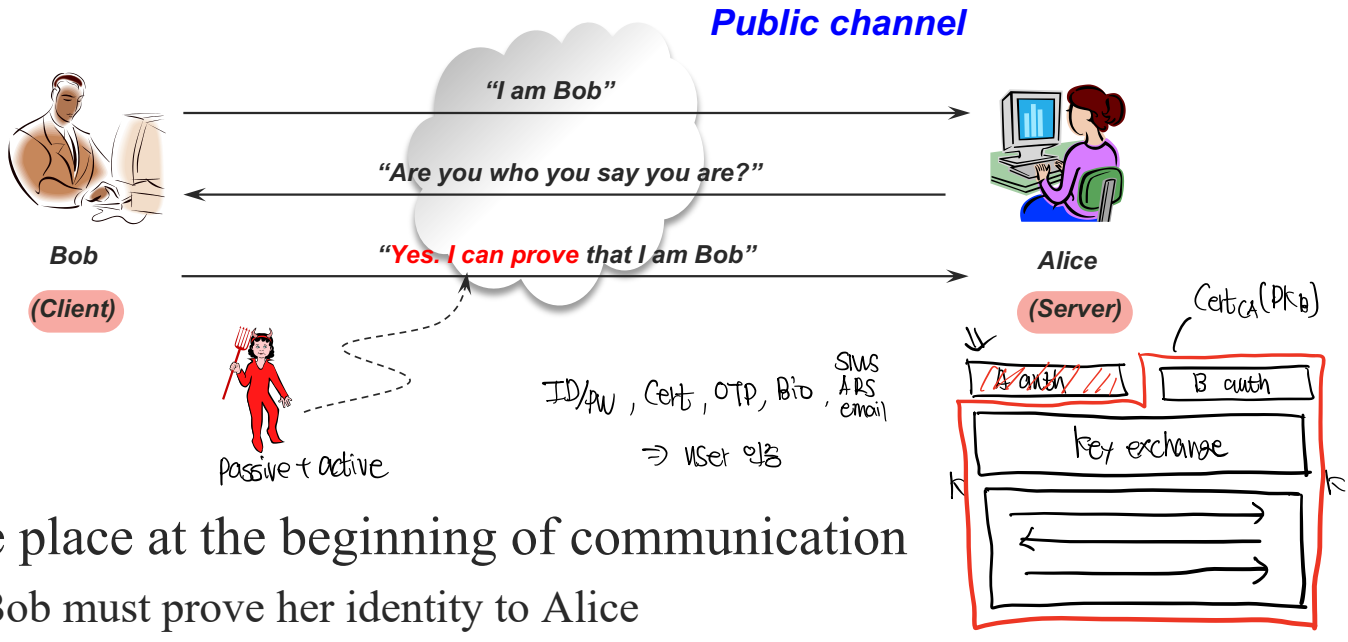
Authentication 사용자 인증

Jong Hwan Park

(Entity) Authentication



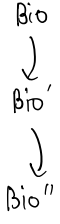
- Take place at the beginning of communication
  - Bob must prove her identity to Alice
  - May also require Alice to prove his identity to Bob (mutual authentication)
  - Authentication over public channel is challenging
    - Attacker can observe or reply messages
    - Active attacks possible (insert, change, impersonate)
  - Provably need to establish a session key (as a simultaneous step)

How to authenticate

- Can be based on
  - Something you know
    - A password, residence registration number, …
  - Something you have
    - A smartcard or security token (storing crypto keys), OTP generator, …
  - Something you are
    - Biometric information like your fingerprint or iris, …
- Can be performed with two-factor authentication
- To perform authentication (and key agreement), need
  - Using symmetric-key / public-key encryption
  - Using message authentication code / digital signature
  - Using hash function / biometric data recognition, …

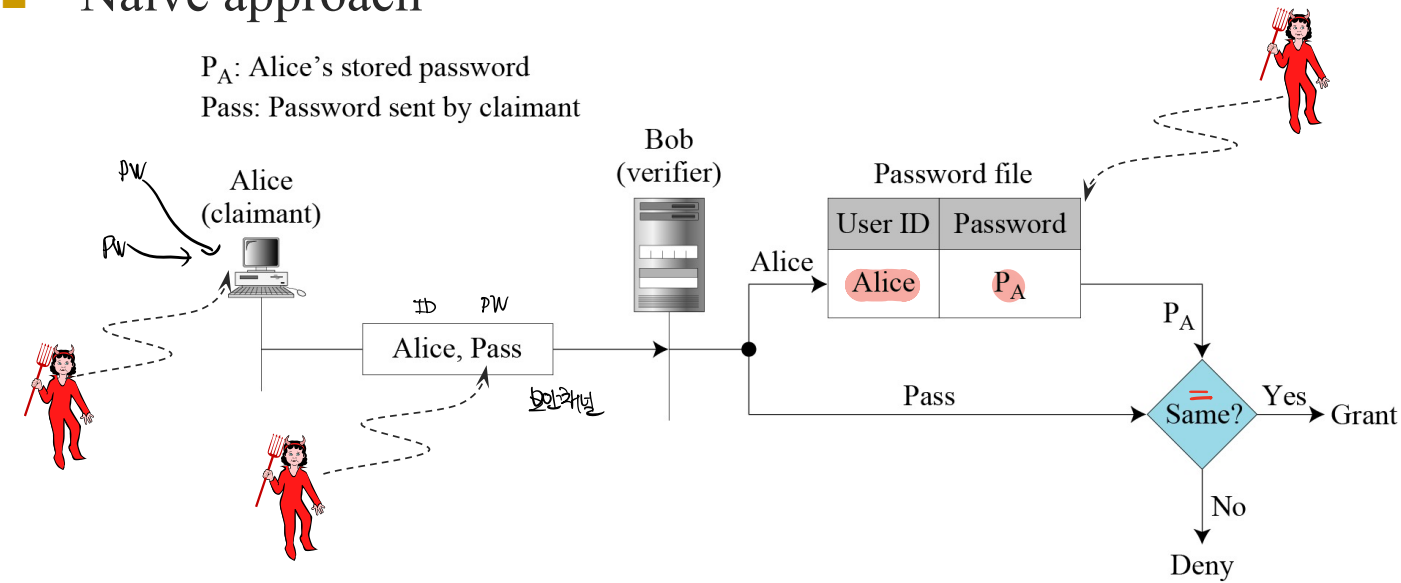
Current Authentication

	ID/PW	Cert	OTP	Bio	OT-CR
Method	password	Certificate + (password)$_{Bio}$	OTP	Biometric data + (others)	Mobile, email
Crypto	Hash	DS	PRF	DS + Recog.	PRNG
HW	No	Smartcard, secure token	OTP device	Secure recog.	No
Others	Keyboard security	(Keyboard security + …)	SW-OTP	Error correction	



Password-based authentication

- Why passwords? (even if password is very weak)
  - Cost(passwords are free) and convenience (easy for server to set up)
- Naïve approach

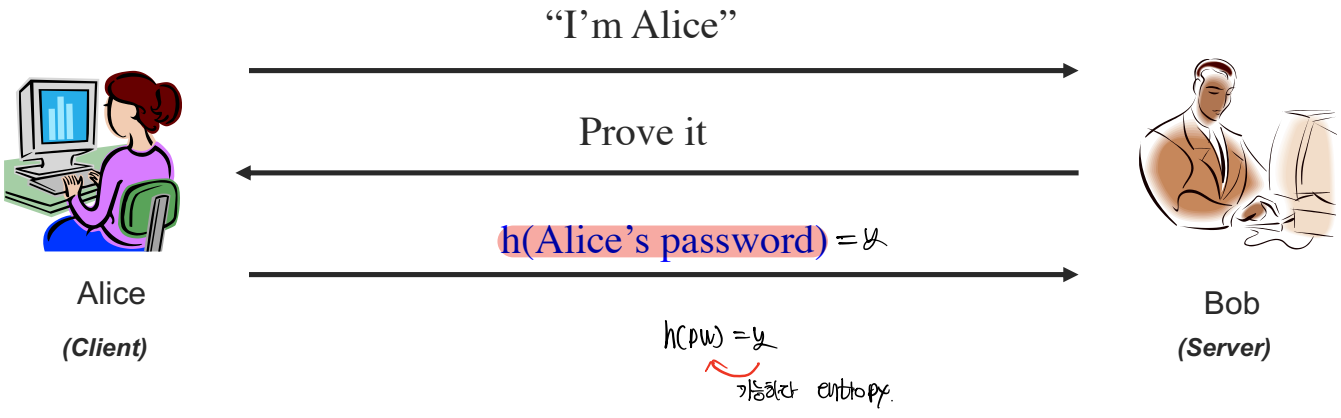


  - Security issues:
    - Password sent to Bob should be protected (due to replay attack!)
    - Password file stored in Bob should also be protected

How to protect PW

client ~~~~~~~~ server
TLS

- A simple example:

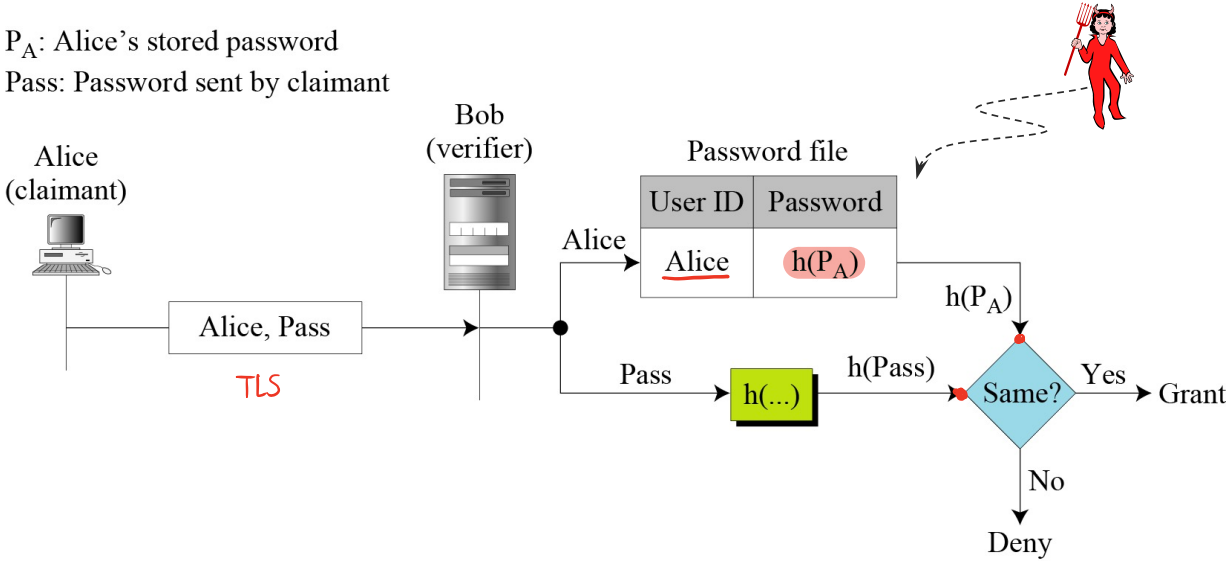


  - Better since h(pw) hides Alice's password from attacker
  - But, still subject to replay attacks

- Need some crypto techniques (see later)
  - Based on challenge-response
    - How can we fix the protocol above against *replay attacks*?

How to protect PW file (1)

- Hashing the passwords and storing them

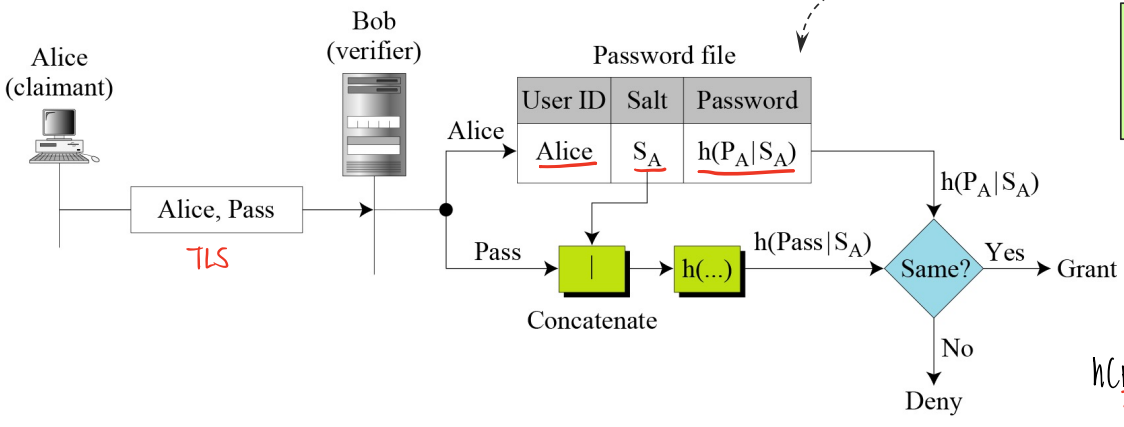


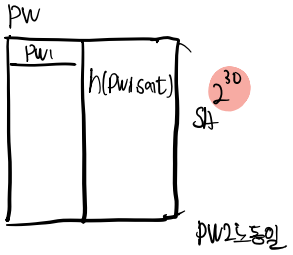
- Assume attacker can access PW file
  - How can attacker find a PW?
    - Pre-compute h(x) for all x in a dictionary of common PWs
    - Compare hashes to its pre-computed dictionary
  - How can we make the dictionary attack more difficult? (→ salt)

How to protect PW file (2)

- Salting the passwords
  - Salt = a random string

P_A: Alice's password
S_A: Alice's salt
Pass: Password sent by claimant





UNIX OS uses salting method

h(pw1 salt) = y

- Assume attacker can access PW file and do dictionary attack
  - If salt is a 30-bit random string, then 2^{30} possible salts
  - If dictionary has 2^{20} passwords, need $2^{20} \cdot 2^{30} = 2^{50}$ hashes and comparison

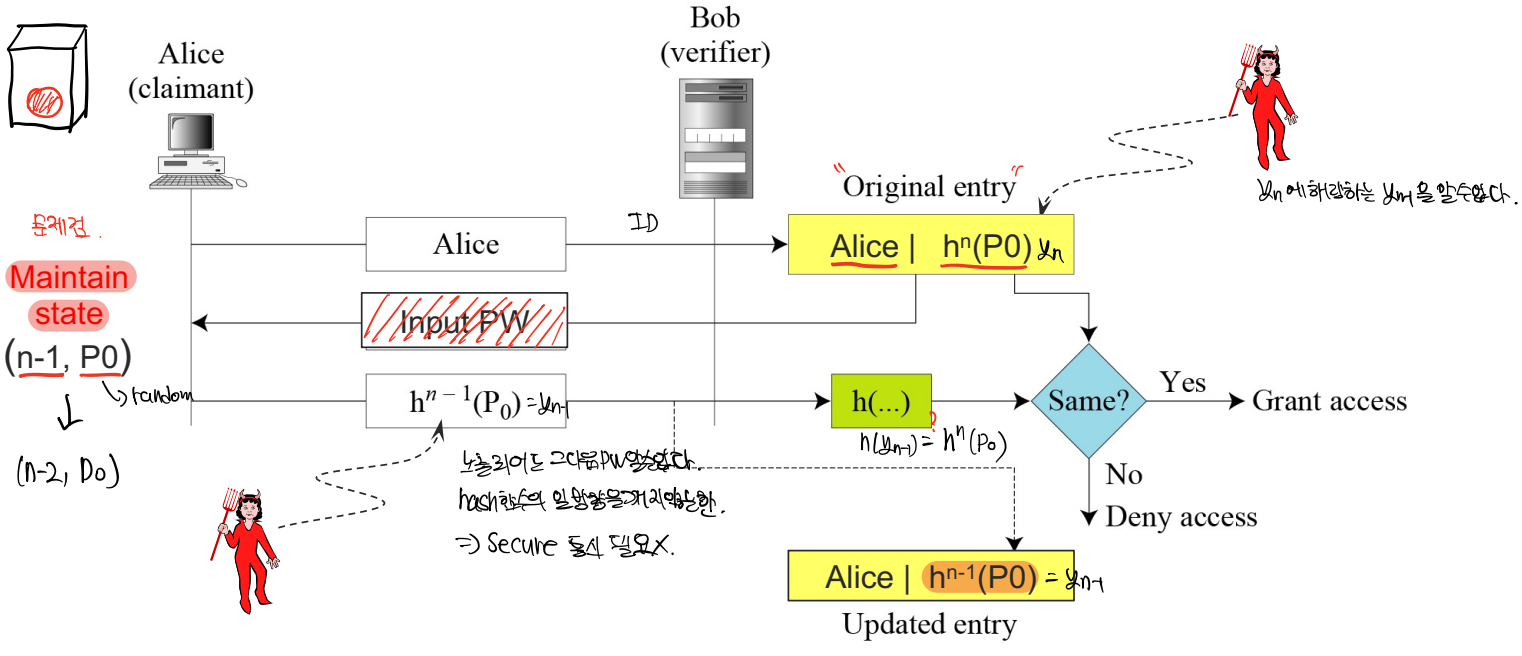
공격을 막을수 없지만 공격 시간을 증가.

One-Time Password (by Lamport)

OTP

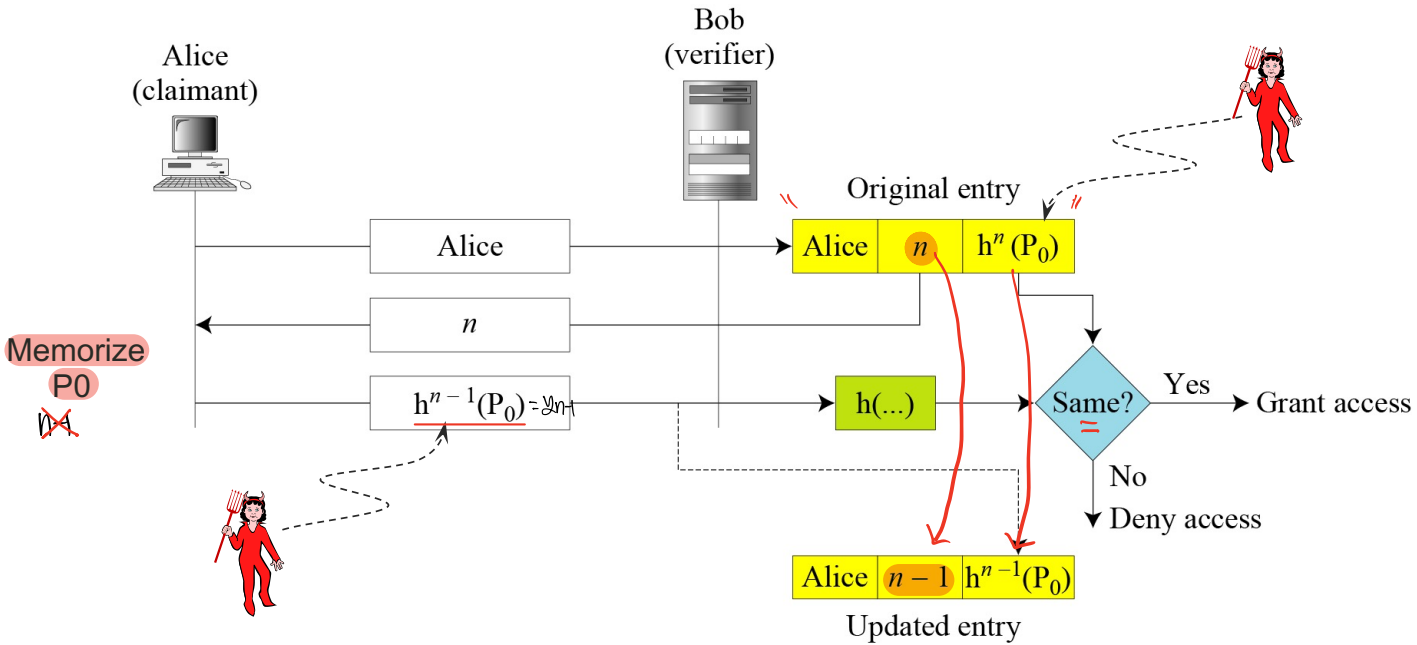
- One-Time Password (OTP) using hash, devised by Lamport
  - Agree on an initial password [Alice, $h^n(P0)$]
  - Agree to sequentially update the password
  - What is the problem ? State 정보를 저장해야한다.

$P_0 \rightarrow h(P_0) = y_1$, $h(y_1) = y_2$, $h(y_{n-2}) = y_{n-1}$, $h(y_{n-1}) = y_n$



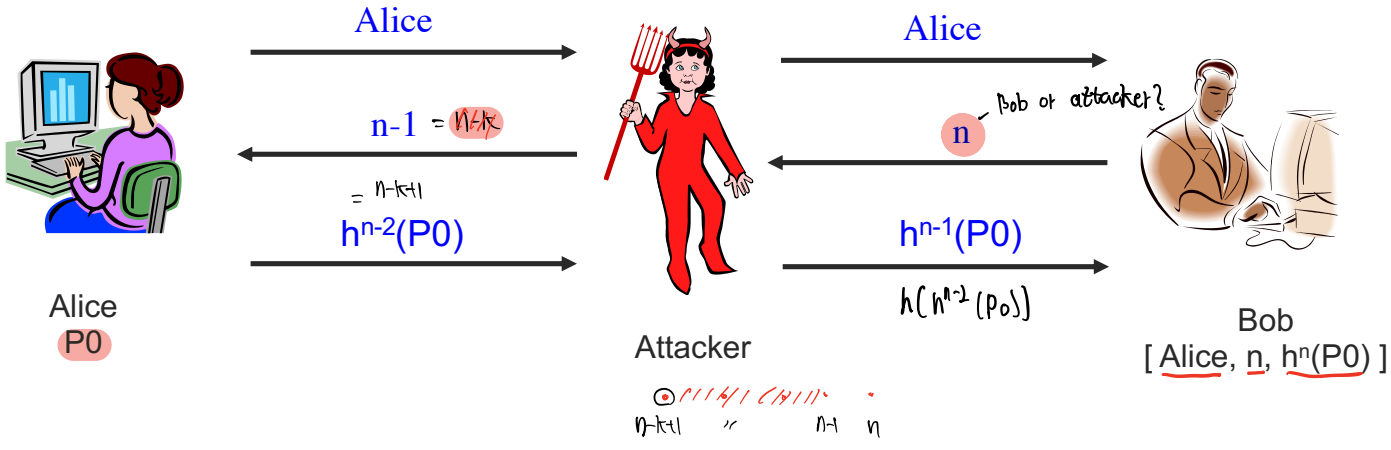
One-Time Password (S/KEY)

- S/KEY: OTP with a flawed modification
  - Agree on an initial password [Alice, n, $h^n(P0)$]
  - No longer lose sync, and unreliable communication link is not problem
  - How can attacker impersonate Alice to Bob?



An Attack on S/KEY

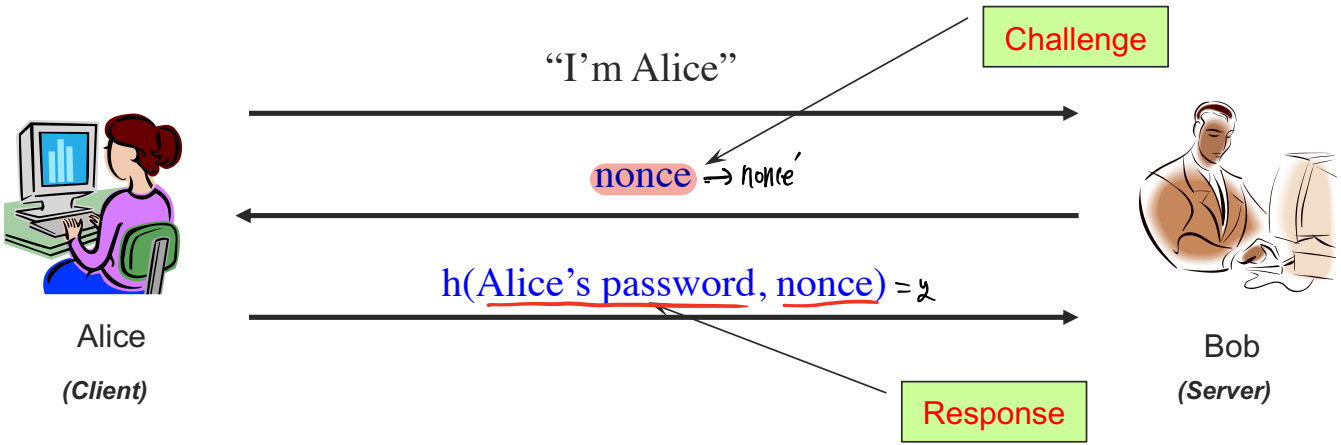
- Man-in-the-middle attack



- Attacker can have $h^{n-2}(P0)$ which he can use for logging-in under the name of Alice in the next section
- What originates the above attack?
  - How to fix it?

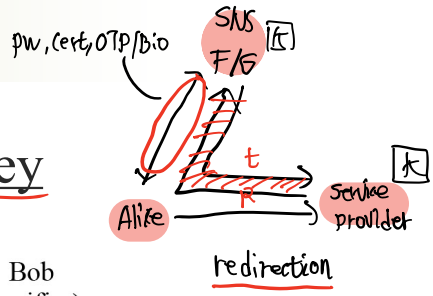
Challenge-Response

- To prevent replay attack, use challenge-response (CR)
  - Goal is to ensure "freshness"
  - To provide freshness, can employ a nonce
    - Nonce = a number used once

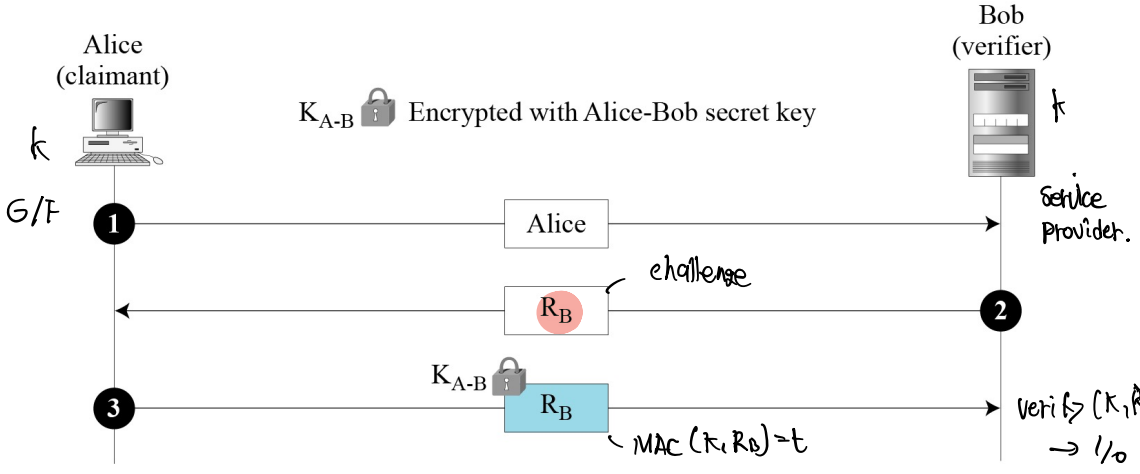


  - Effects:
    - Reply is not possible
    - Only Alice can provide the correct response
    - Alice can prove she knows a secret without sending it to Bob

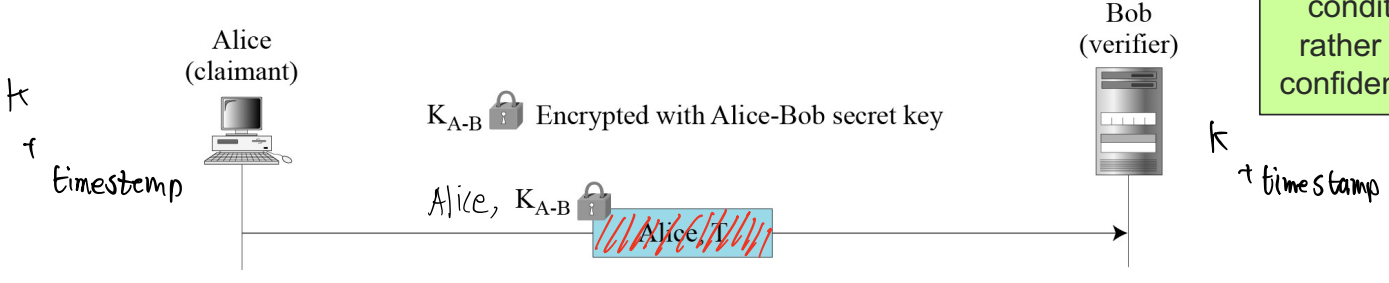
CR Authentication using SKE (1)



- Assume Alice and Bob know a shared secret key
  - Alice is authenticated to Bob



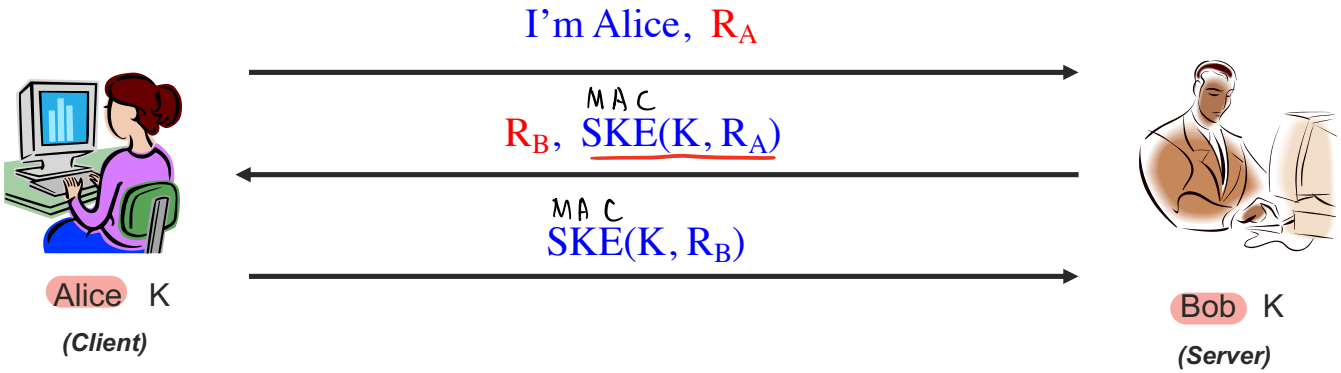
  - In case of timestamp challenge



Integrity is a necessary condition, rather than confidentiality

CR Authentication using SKE (2)

- Mutual authentication
  - In previous one-way CR protocol, Bob can authenticate Alice
  - But, Alice cannot authenticate Bob (due to "phishing server")
  - A (misguided) protocol
    - By simply employing one-way CR protocol twice
    - Once for Alice and once for Bob



  - Is it secure or not? Why is that?

CR Authentication using SKE (3)

- Attack on the (misguided) mutual authentication protocol

Attacker

1. "I'm Alice", R_A

2. R_B, $E(R_A, K)$ ✓

5. $E(R_B, K)$ ✓

Bob K

Attacker

3. "I'm Alice", R_B

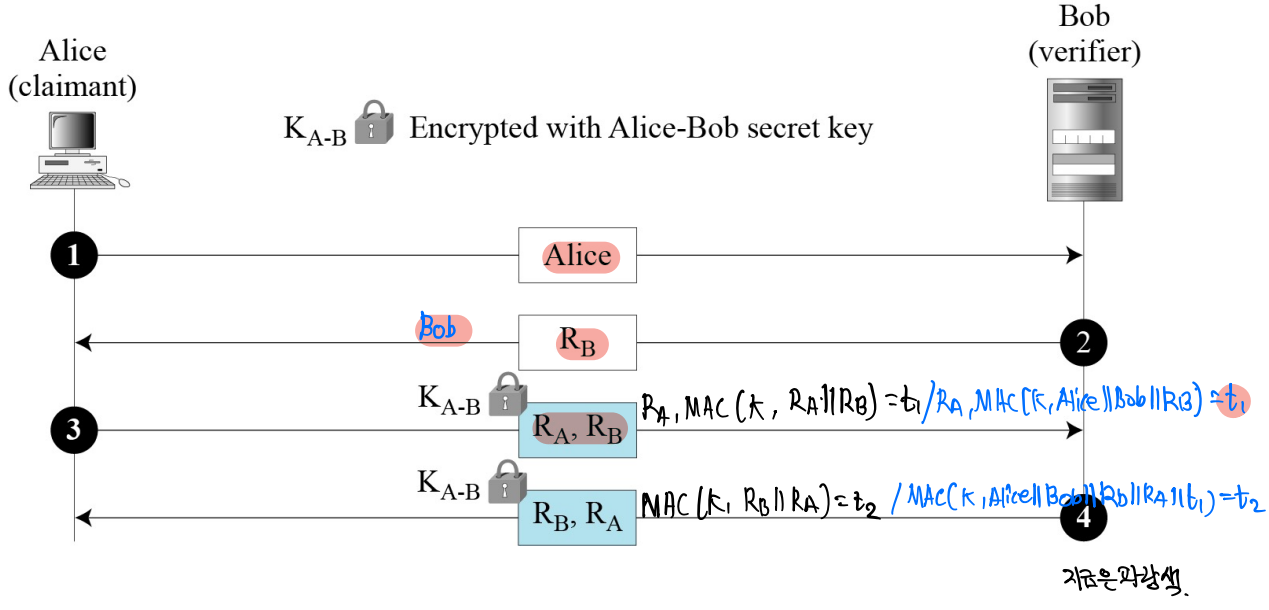
4. R_C, $E(R_B, K)$

Bob K

How can we fix it?

CR Authentication using SKE (4)

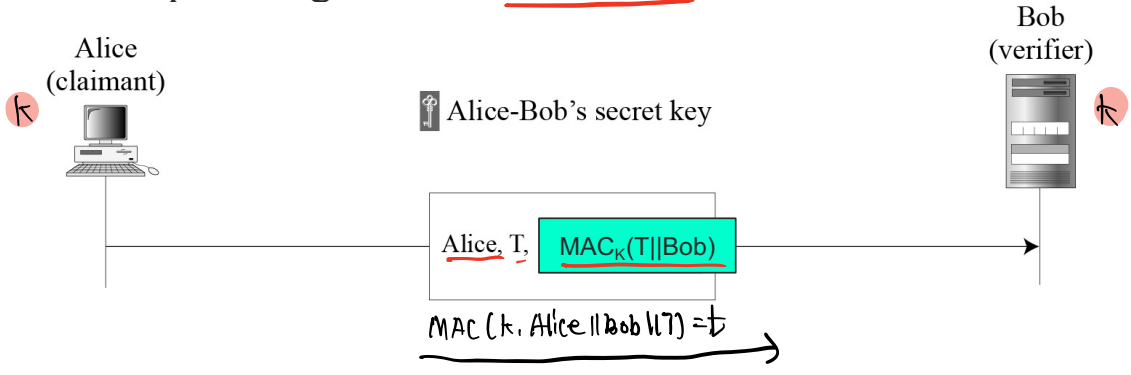
- Mutual authentication 버대칭암호 필요하다.
  - Assume Alice and Bob share a secret key



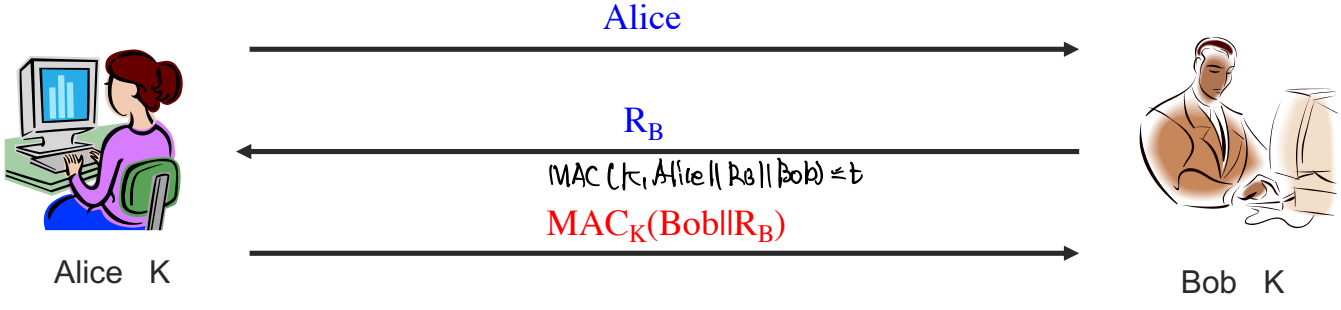
  - In the (4), the order of R_A and R_B are switched
    - Otherwise, it can prove nothing about the sender (bob) (why?)
    - More desirable to include 'Bob' and 'Alice' in (3) and (4) messages, resp.

CR Authentication using MAC (1)

- Assume Alice and Bob shares a secret key
  - Advantage is to preserve integrity of challenge and response messages
  - An example using MAC + timestamp:

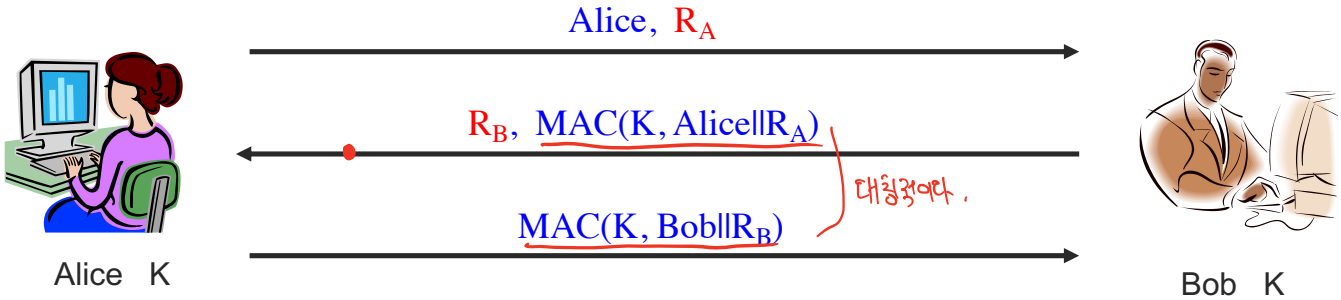


  - An example using MAC + nonce:



CR Authentication using MAC (2)

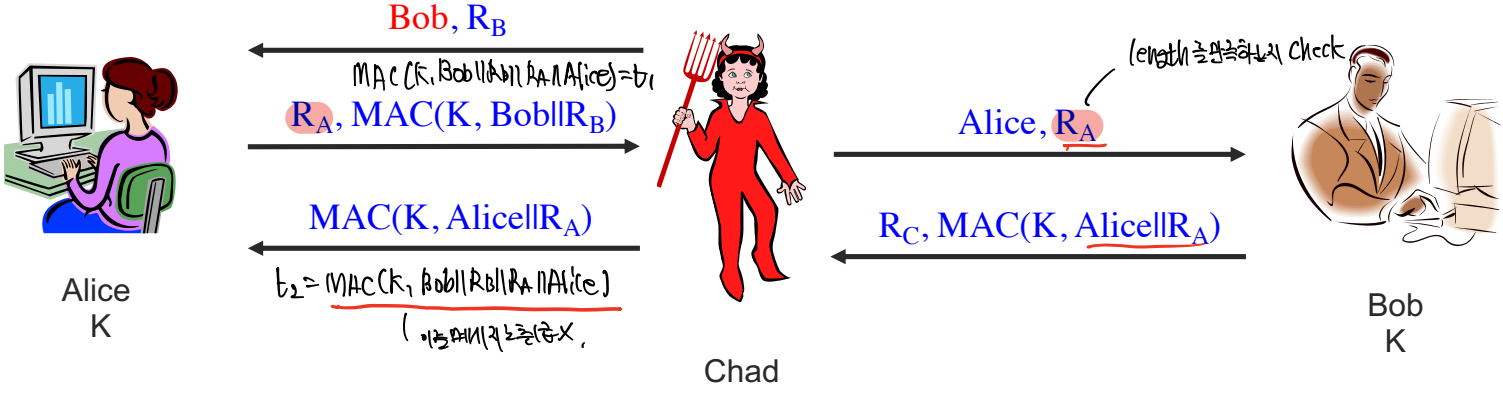
- Assume Alice and Bob shares a secret key
  - How about mutual authentication? Is it secure?



  - Man-In-the-Middle (MIM) attack occurs. How can we fix it?

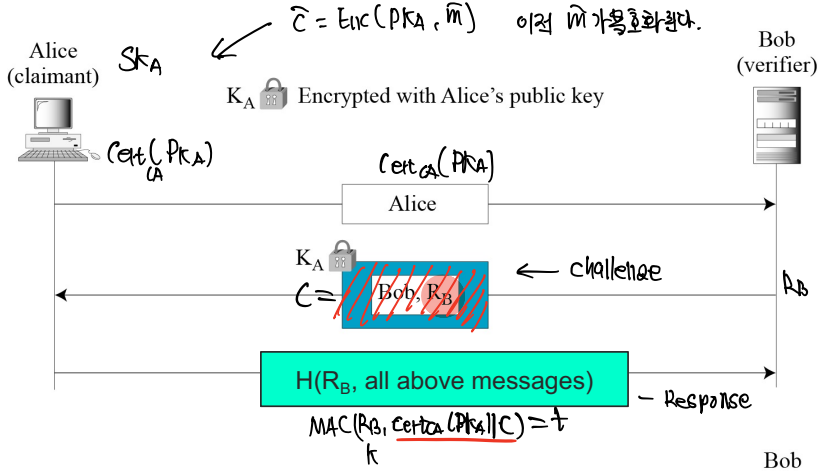
An Attack

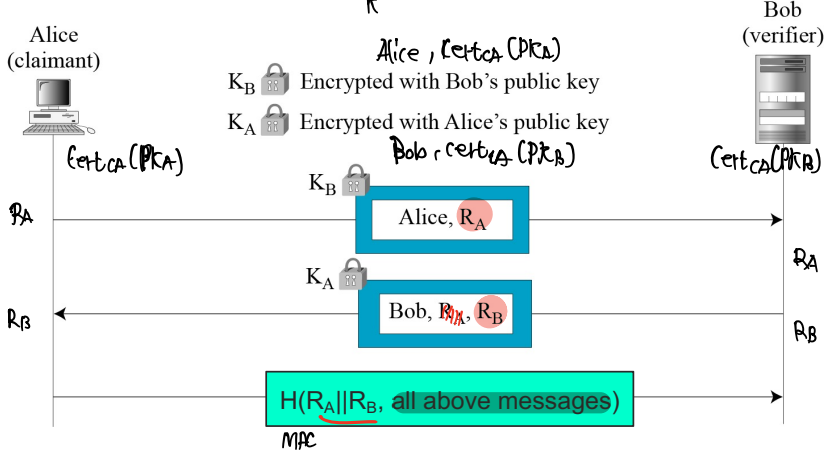
- MIM attack
  - Assume Alice and Bob share a symmetric-key K
  - Chad impersonates Bob to Alice



CR Authentication using PKE

- Assume Alice owns SK_A corresponding to PK_A
  - One-way authentication
    - Alice → Bob
  - What is the problem?
    - Get Alice to decrypt any CT
    - Should have different keys
    - How to fix it?
  - Mutual authentication
    - Bob also has (PK_B, SK_B)
    - Same problem to above



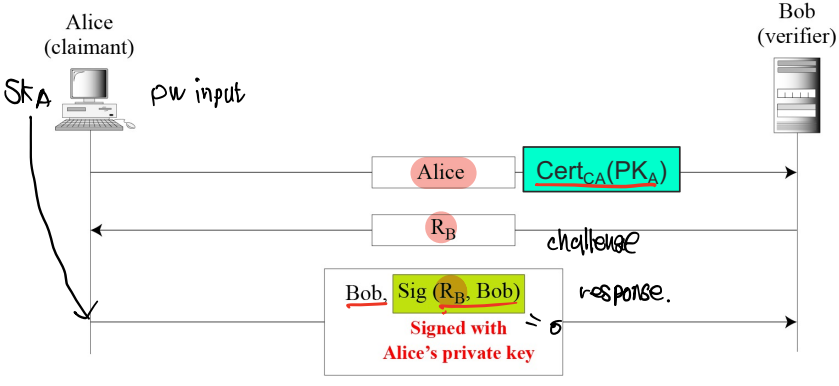


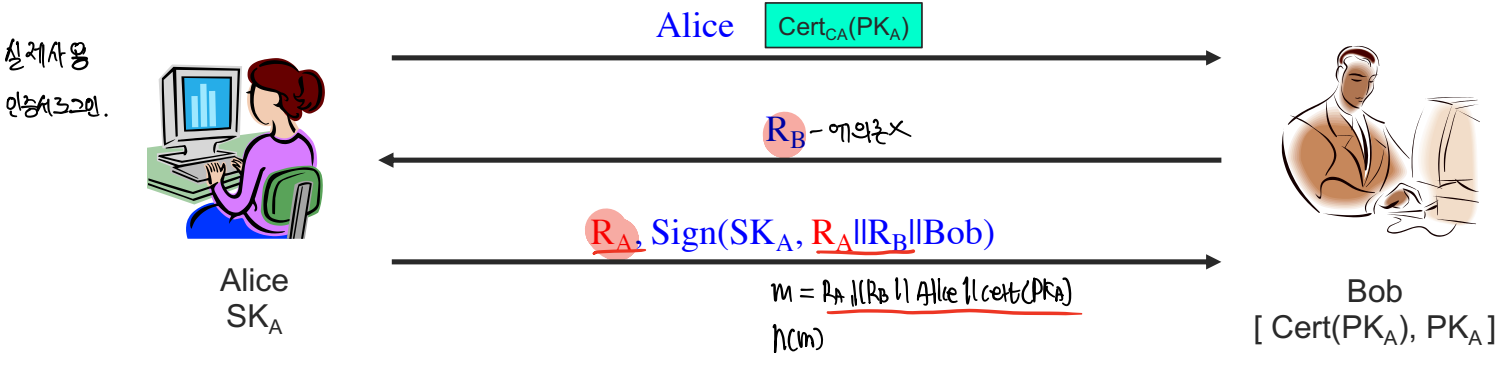
In practice, Cert(PK_A) or Cert(PK_B) should be sent to other party

In certificates, the usage of public keys can be specified

CR Authentication using DSS (1)

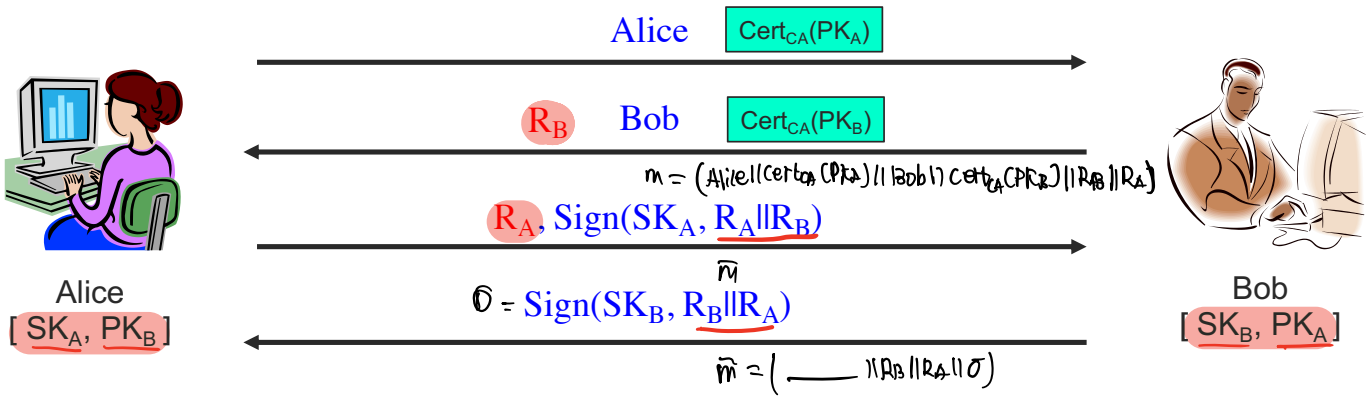
- Assume Alice owns SK_A corresponding to PK_A
  - One-way authentication
    - Alice → Bob
  - Cert(PK_A) is sent to Bob
  - What is the problem?
    - Recall `hash-then-sign'
    - Bob may have prepared R_B as R_B=hash(Transfer $10^7$$ from Alice to Bob)
  - How to fix it:





CR Authentication using DSS (2)

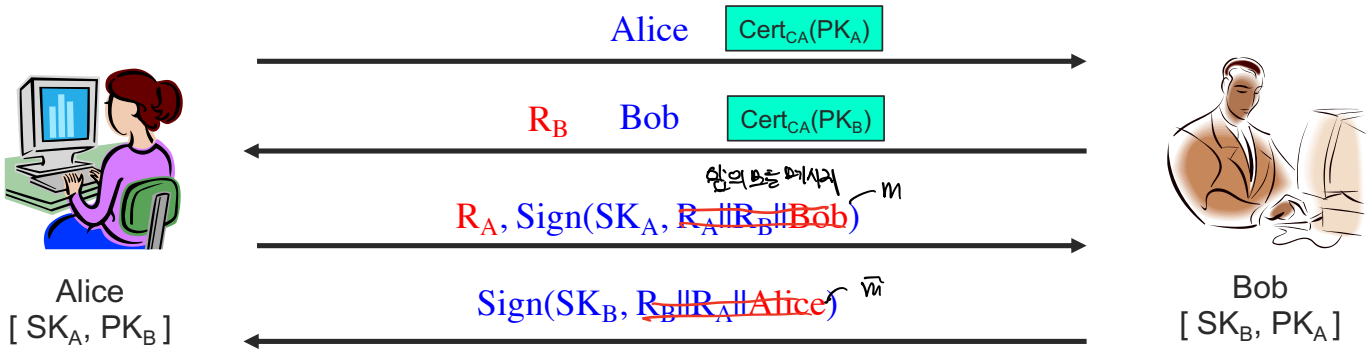
- Mutual authentication
  - Assume Alice an Bob have $[PK_A, SK_A]$ and $[PK_B, SK_B]$, resp.
  - One (misguided) protocol:



  - Is the above protocol secure?
  - How can an adversary impersonate Bob to Alice?
    - Assume the adversary (Chad) has its own $[SK_C, PK_C]$

CR Authentication using DSS (3)

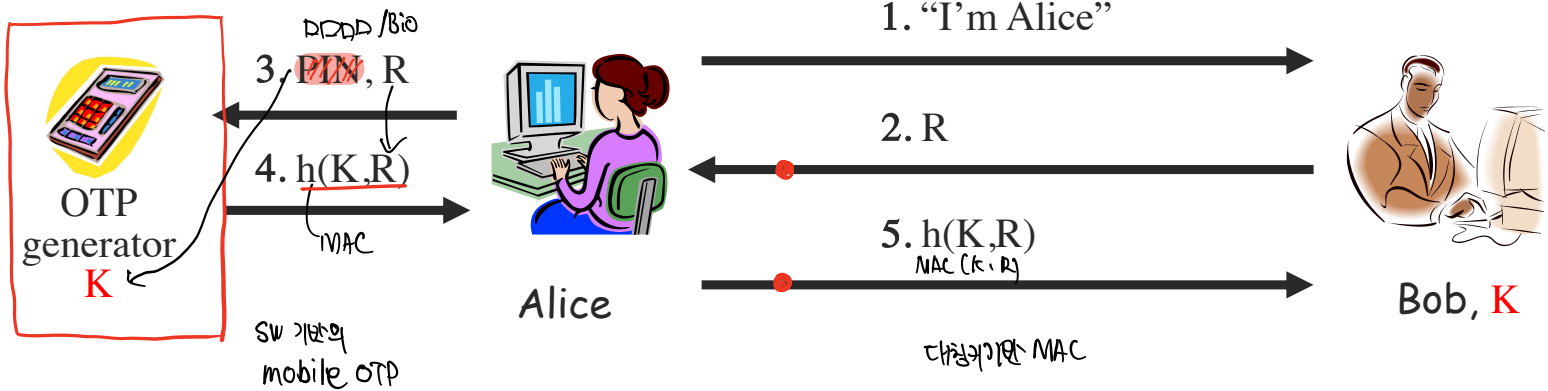
- Mutual authentication
  - Assume Alice an Bob have $[PK_A, SK_A]$ and $[PK_B, SK_B]$, resp.
  - One protocol:



  - What is the difference between the previous protocol and this one?
  - Meaning that authentication protocol has error-prone nature, although crypt primitives used are secure

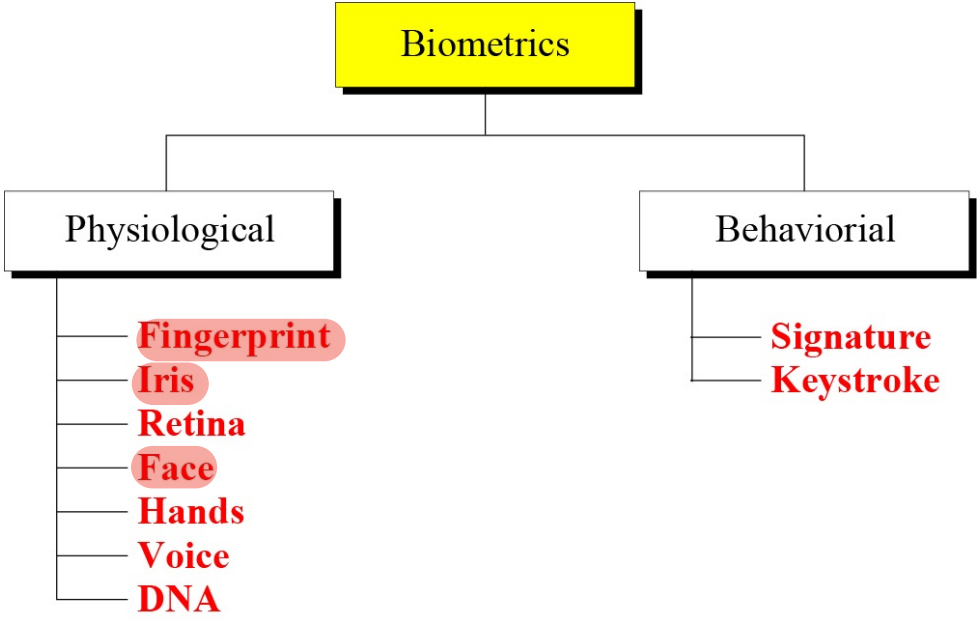
One-Time Password: Something you have

- How OTPs are generated:
  - (1) Based on time-synchronization (in a short period of time)
  - (2) Based on previous OTP values (like ~~S/KEY~~)
  - (3) Based on challenge-response
- Approaches (1) and (3) need OTP-generator
  - That stores a symmetric-key K shared between Alice and Bob
  - That can perform simple operations like hash and AES



Authentication: Something you are

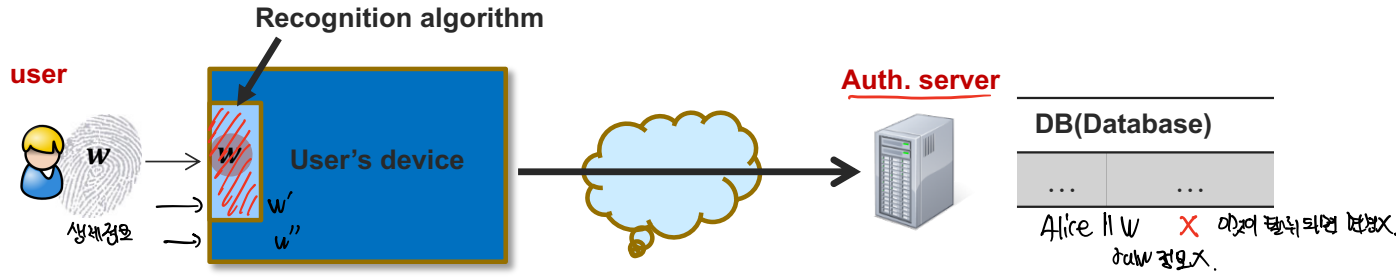
- Something you are - Biometrics



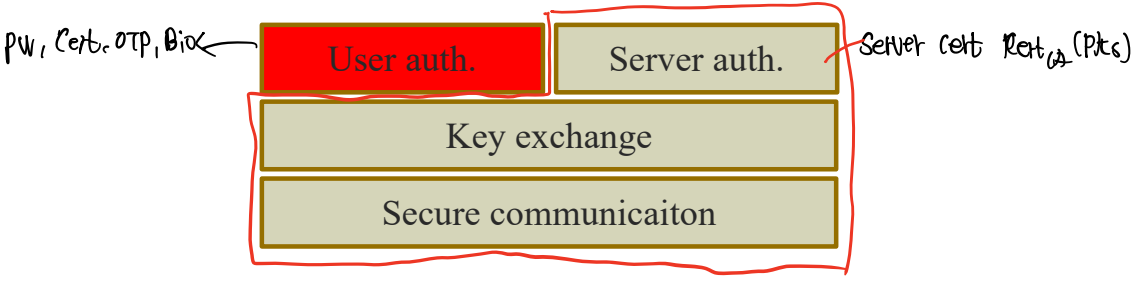
- Several applications
  - Access to facilities, investigations, forensic analysis, immigration control

Biometric Authentication (1)

- Basic facts

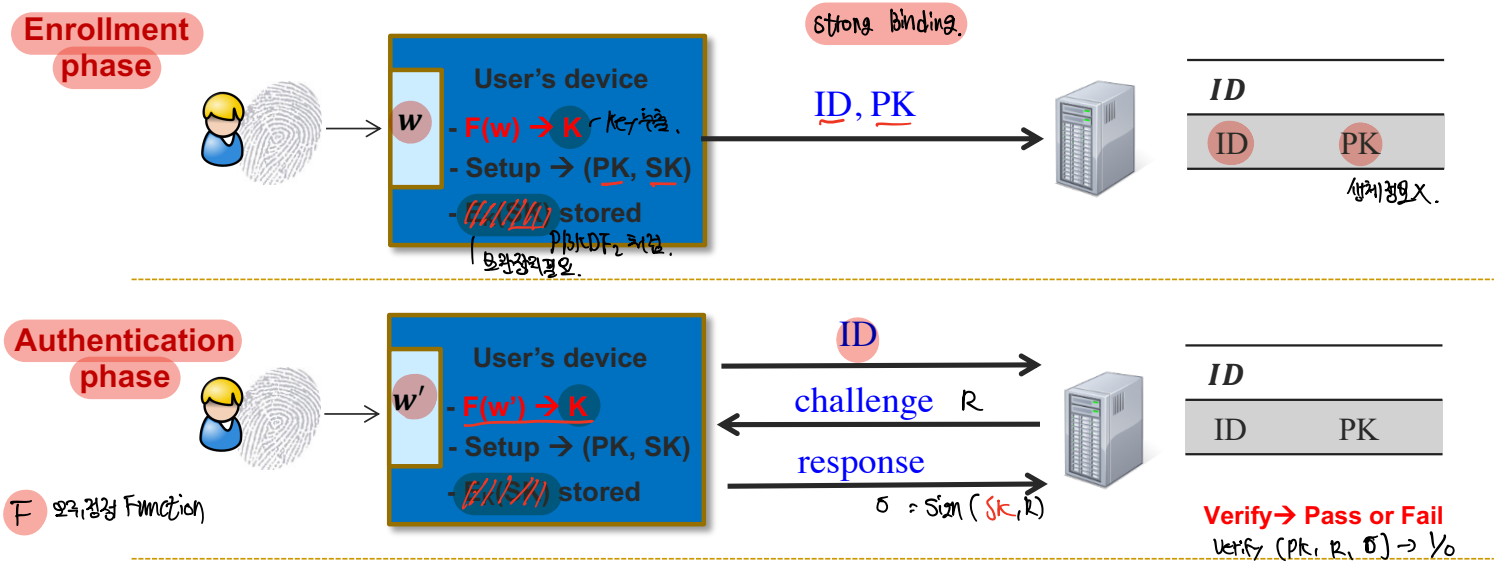


  - Only user authentication (user → auth. server)
  - Not considered server auth. (user ← auth. Server)
    - Using SSL/TLS, server authentication/key exchange/secure communication can be done by $Cert_{CA}(PK_{server})$



Biometric Authentication (2)

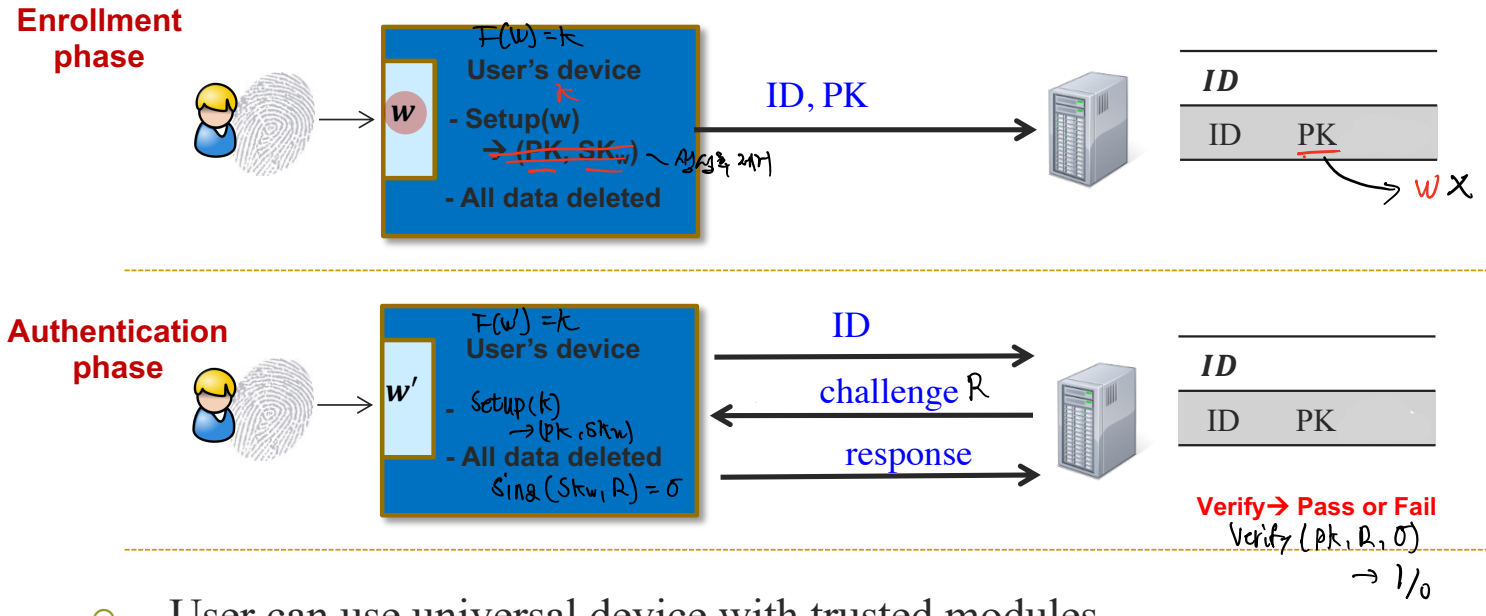
- Basic process of bio. Authentication (device-centric model)



- Assume user should have device with secure storage
- w, w': biometric data may contain errors in each measurement process
- User auth. is passed when an error occurs only if $dist(w, w') \leq t$

Biometric Authentication (3)

- Basic process of bio. Authentication (universal device model)

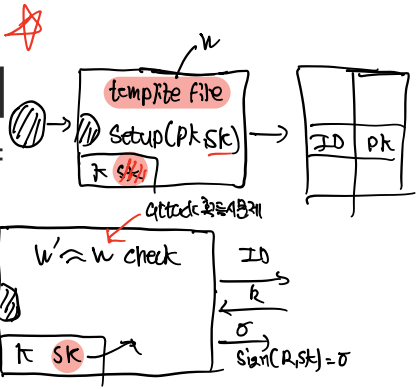


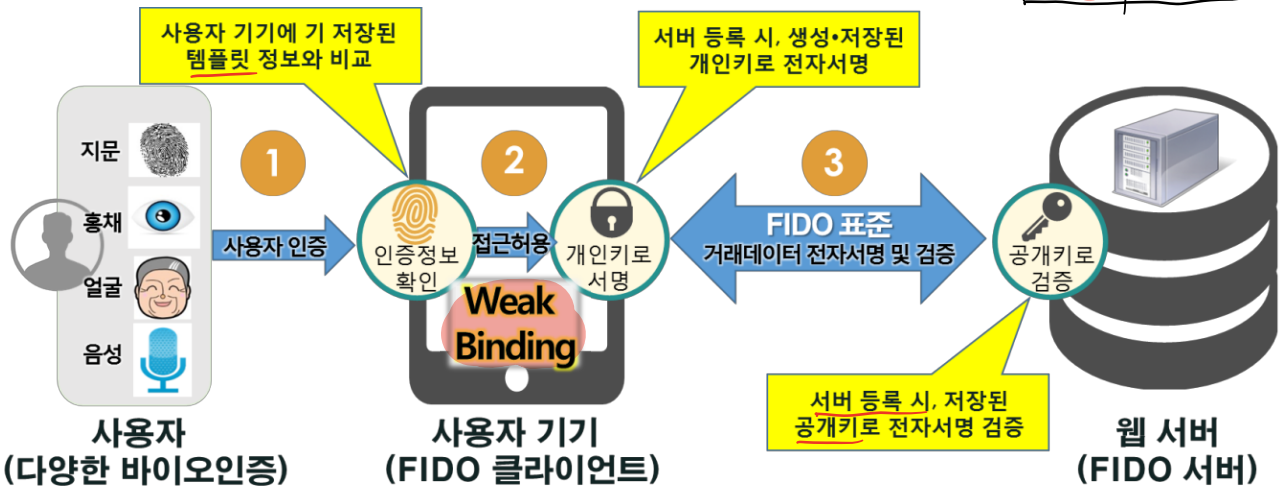
  - User can use universal device with trusted modules
  - User auth. is passed when an error occurs only if $dist(w, w') \leq t$
  - In verification, outputs correct value

FIDO (1) – key idea

Fast IDentity Online ⇒ raw data와 비교에 저장

- 사용자 인증과 원격 인증 프로토콜의 분리
  - 사용자 인증수단은 바이오, 토큰, 패턴 등 다양
  - 원격 인증은 공개키 기반 단일 방식
    - 서버 변경 없이 다양한 인증 수단 사용

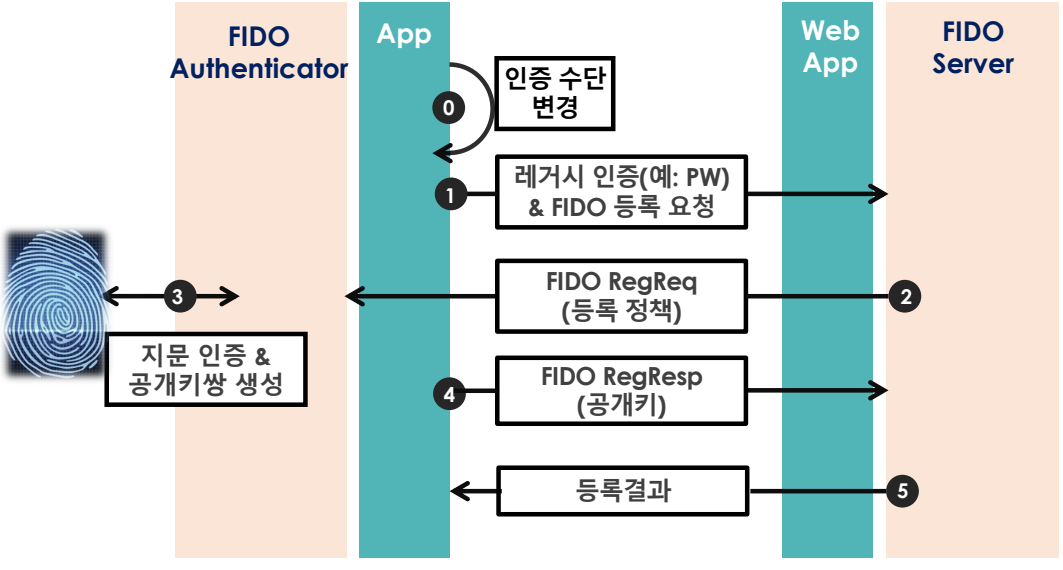




사용자 → w ⟷ SK → σ
(w와 SK 존재가 약하다.)

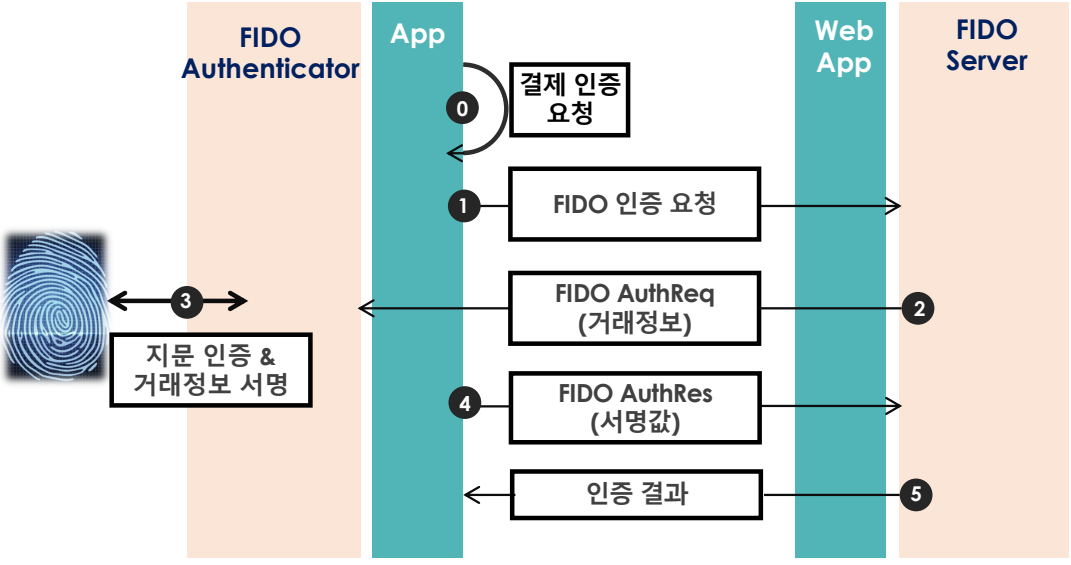
FIDO (2)

- FIDO registration protocol



FIDO (3)

- FIDO authentication protocol



FIDO (4) - privacy

- 사용자 인증정보는 사용자 소유 기기에만 저장
  - 민감한 바이오 정보는 사용자 로컬 인증을 수행하는 사용자 기기에만 저장되고 사용
  - 인증 서버는 원격 인증을 위한 랜덤 공개키 정보만 저장
- 사이트에는 서로 다른 공개키 정보를 등록
  - 등록 과정에서 사이트 별로 랜덤하게 생성하여 등록
  - 서버 데이터의 대규모 유출에 대응 : 다른 사이트에서 재사용 불가
  - 공개키 정보의 손쉬운 재 등록 (예: FIDO 탈퇴 및 등록을 동시에 수행)

Best Authentication Protocol?

- Authentication using SKE, MAC, PKE, DSS needs devices that store users' secret keys
  - Based on "Something you have"
  - Smartcard, secure token, mobile phone, OTP generator

- It depends on…
  - The sensitivity of application/data
  - The delay that is tolerable
  - The cost of keeping storage devices secret
  - The cost(computation) that is tolerable
  - What crypto is supported (PKE, DSS, SKE,…)
  - Whether mutual authentication is required
  - Whether forward secrecy, anonymity, etc., are concern
- … and possibly collaborated with other factors

Q & A